Amendments to the Claims:

This listing of claims will replace all prior versions, and listings, of claims in the application:

Listing of Claims:

1. (Currently Amended) A method for verifying user memory validity in an OS (Operating System), comprising:

generating a system call for a user process;

declaring areas containing certain functions as a safeguard function area;

verifying a validity of a user buffer address area using a user buffer address checking function ~~declared as the safeguard~~;

if the user buffer address area is not valid, determining whether the user buffer address checking function is ~~declared as~~ a function in the safeguard function area by calling an exception processor~~, if the user buffer address area is not valid~~;

if the user buffer address checking function is identified as a function in the safeguard function area, identifying ~~an~~ a safeguard function area identifier ~~of the safeguard~~ by calling a safeguard exception processor~~, if the user buffer address checking function is identified as a function in the safeguard area~~;

identifying whether the user buffer address checking function is defined in the system by identifying the safeguard function area identifier; and

returning an error value to the user processor if the user buffer address checking function is defined in the system.

2. (Currently Amended) The method according to claim 1, further comprising returning a success value to the user processor, and at the same time, ~~exiting~~ revoking the ~~declared~~ safeguard function declaration if the user ~~memory~~ buffer address area is valid.

3. (Original) The method according to claim 1, further comprising exiting the user processor if the user buffer address checking function is not defined in the system.

4. (Currently Amended) The method according to claim 1, wherein ~~the step of~~ verifying the validity of a user buffer address area comprises:

detecting a page within the user buffer address area; and

determining whether a fault is generated by sequential access (read/write) to the address area of the detected page.

5. (Currently Amended) The method according to claim 4, wherein ~~the step of~~ verifying the validity of a user buffer address area further comprises accessing via a Kernal the user buffer address area and verifying the validity of the user buffer.

6. (Original) The method according to claim 5, wherein the Kernel processes a fault generated in the Kernel area as a simple error.

7. (Currently Amended) The method according to claim 5, wherein the Kernel uses a safeguard function to process the fault generated in the Kernel area.

8. (Currently Amended) The method according to claim 1, wherein ~~the step of~~ verifying the validity of a user buffer address area is performed without using a MMU (Memory Management Unit) Table.

9. (Currently Amended) The method according to claim 1, wherein each area declared as a safeguard function area has a unique identifier.

10. (Currently Amended) A method for verifying user memory validity in an operating system (OS), comprising:

performing a system call for a user process;

declaring a validity checking function as a safeguard function in a safeguard function area;

determining whether a user memory area is valid using the validity checking function;

if the user memory area is not valid, identifying whether the validity checking function is declared as the safeguard function by calling an exception processor ~~if the user memory area is not valid~~;

if the validity checking function is in the safeguard function area, calling a safeguard exception processor; ~~and~~

identifying an identifier of the safeguard function area ~~exception processor if the validity checking function is in the safeguard function area~~;

recognizing via the safeguard function area identifier ~~exception processor~~ that a subject of the process is the validity checking function and identifying whether the validity checking ~~the~~ function is defined in the system ~~through the identifier of the safeguard function~~; and

if the validity checking function is defined in the system, returning an error to the user processor ~~processing the validity checking function as defined in the system which performs the process of the function, if the validity checking function is defined in the system~~.

11. (Currently Amended) The method according to claim 1, further comprising ~~verifying a validity of a user memory area using the validity checking function, wherein the method further comprises~~ returning a success value to the user processor, and at the same time, ~~exiting~~ revoking the ~~declared~~ safeguard function declaration, if the user memory area is valid.

12. (Original) The method according to claim 11, further comprising exiting the user processor when the validity checking function is not defined in the system.

13. (Currently Amended) The method according to claim 12, further comprising accessing via a Kernal a user memory area and verifying a validity of the user ~~buffer~~ memory.

14. (Original) The method according to claim 13, wherein the Kernel treats a fault generated in the Kernel area as a simple error.

15. (Original) The method according to claim 13, wherein the Kernel uses the safeguard function in order to process the fault generated in the Kernel area.

16. (Original) The method according to claim 10, wherein the validity verifying step is executed without using a MMU (Memory Management Unit) Table.

17. (Currently Amended) The method according to claim 10, wherein ~~the~~ each declared safeguard function area ~~include~~ includes a unique identifier.

18. (Currently Amended) A computer-readable medium having stored thereon a sequence of instructions which, when executed by a processor, cause the processor to at least perform a method comprising:

generating a system call for a user process;

declaring areas containing certain functions as a safeguard ~~function~~ area;

verifying validity of a user buffer address area using a user buffer address checking function ~~declared as the safeguard function~~;

if the user buffer address area is not valid, determining whether the user buffer address checking function is declared as a function in the safeguard function area by calling an exception processor~~, if the user buffer address area is not valid~~;

if the user buffer address checking function is identified as a function in the safeguard area, identifying a safeguard ~~establishing an~~ identifier of the safeguard function area by calling a safeguard exception processor~~, if the user buffer address checking function is identified as a function in the safeguard area~~;

confirming whether the user buffer address checking function is defined in the system by identifying the safeguard function area identifier; and

returning an error value to the user processor if the user buffer address checking function is defined in the system.

19. (Currently Amended) The computer-readable medium of claim 18, wherein the sequence of instructions further causes the processor to perform the step of returning a success value to the user processor, and at the same time, ~~exiting~~ revoking the ~~declared~~ safeguard function declaration if the user memory is valid.

20. (Currently Amended) The computer-readable medium of claim 18, wherein the sequence of instructions further causes the processor to perform the step of exiting the user processor if the user buffer address checking function is not defined in the system.

21. (Currently Amended) The computer-readable medium of claim 18, wherein the sequence of instructions further causes the processor to perform the steps of:

detecting a page within the user buffer address area; and

determining whether a fault is generated by performing at least one read/write function to the address area of the detected page.

22. (Currently Amended) The computer-readable medium of claim 21, wherein the step of verifying validity comprises accessing via a Kernel the user buffer address area and verifying the validity of the user buffer.

23. (Currently Amended) The computer-readable medium of claim 22, wherein the step of verifying validity further comprises processing via the Kernel a fault generated in the Kernel area as a simple error.

24. (Currently Amended) The computer-readable medium of claim 22, wherein the step of verifying validity further comprises using within the Kernel a safeguard function to process the fault generated in the ~~kernel~~ Kernal area.

25. (Currently Amended) The computer-readable medium of claim 18, wherein the step of verifying the validity of a user buffer address area is performed without using a memory management table.

26. (Currently Amended) The computer-readable medium of claim 18, wherein each ~~respective~~ area declared as a safeguard function area has a unique identifier.

27. (New) The method according to claim 1, wherein declaring areas containing certain functions as a safeguard function area is performed in response to the system call.

28. (New) The method according to claim 10, wherein declaring a validity checking function as a safeguard function in a safeguard function area is performed in response to the system call.

29. (New) The method according to claim 18, wherein declaring areas containing certain functions as a safeguard area is performed in response to the system call.